

What is claimed is:

1 1. An apparatus comprising:

2 marking logic to mark instruction information for an instruction of a speculative thread as
3 speculative; and

4 blocker logic to prevent data associated with a store instruction of the speculative thread
5 from being forwarded to an instruction of a non-speculative thread, the blocker logic further
6 to prevent the data from being stored in a memory system.

1 2. The apparatus of claim 1, wherein:

2 blocker logic is further to allow the data associated with a store instruction of the speculative
3 thread to be forwarded to an instruction of a second speculative thread.

1 3. The apparatus of claim 1, further comprising:

2 a plurality of store request buffers, each store request buffer including a speculation
3 identifier field.

1 4. The apparatus of claim 1, wherein the memory system further comprises:

2 a data cache that includes a safe-store indicator field associated with each entry of a tag
3 array.

1 5. The apparatus of claim 1, wherein:
2 the blocker logic is included within the memory system.

1 6. The apparatus of claim 1, wherein blocker logic further includes:
2 dependence blocker logic to prevent data associated with a speculative store instruction
3 from being forwarded to an instruction of the non-speculative thread ; and
4 store blocker logic to prevent the data from being stored in a memory system.

1 7. The apparatus of claim 6, wherein:
2 store blocker logic is outside the execution pipeline.

1 8. The apparatus of claim 7, wherein:
2 store blocker logic is included in the memory system.

1 9. The apparatus of claim 6, wherein:
2 dependence blocker logic is included in an execution pipeline.

1 10. The apparatus of claim 9, wherein:
2 dependence blocker logic is included in a memory ordering buffer.

1 11. A system, comprising:

2 a memory system that includes a dynamic random access memory;

3 a processor including dependence blocker logic to prevent data associated with a store
4 instruction of a speculative thread from being forwarded to an instruction of a non-
5 speculative thread;

6 the processor further including store blocker logic to prevent the data from being stored
7 in the memory system.

1 12. The system of claim 11, wherein:

2 the processor further includes marking logic to mark instruction information associated
3 with the store instruction as speculative.

1 13. The system of claim 12, wherein:

2 the marking logic is further to associate a safe speculation domain ID with the instruction
3 information.

1 14. The system of claim 13, wherein:

2 the marking logic is further to indicate a thread identifier as the speculation domain ID.

1 15. The system of claim 12, further comprising:

2 a store request buffer to store the speculation domain ID.

1 16. The system of claim 11, wherein:

2 the processor includes a first logical processor to execute the non-speculative thread; and

3 the processor includes a second logical processor to execute the speculative thread.

1 17. The system of claim 11, further comprising:

2 a second processor that includes said dependence blocker logic and said store blocker

3 logic;

4 wherein said processor is to execute the non-speculative thread and said second processor

5 is to execute the speculative thread.

1 18. The system of claim of claim 11, wherein:

2 the memory system includes a cache organized to include a plurality of tag lines, wherein

3 each tag line of the cache includes a unique helper thread ID field.

1 19. The system of claim 11, wherein:

2 the memory system includes a cache organized to include a plurality of tag lines, wherein

3 each tag line of the cache includes a safe-store indicator field.

1 20. The system of claim 11, wherein:

2 the memory system includes a victim tag cache to indicate evicted cache lines that
3 include speculative load data.

1 21. A method, comprising:

2 receiving instruction information for a load instruction, the instruction information
3 including a load address;

4 performing a dependence check, wherein performing the dependence check includes:

5 determining if a store address of an in-flight store instruction matches the load
6 address; and

7 determining if the load instruction and the in-flight store instruction each originate
8 with a speculative thread;

9 forwarding, if the dependence check is successful, store data associated with the in-flight
10 store instruction to the load instruction; and

11 declining to forward, if the dependence check is not successful, the store data to the load
12 instruction.

1 22. The method of claim 21, wherein performing the dependence check further
2 comprises:

3 determining if the in-flight store instruction and the load instruction originate from the
4 same thread.

1 23. The method of claim 22, wherein determining if the in-flight store instruction and
2 the load instruction originate from the same thread further comprises:

3 determining if a thread ID associated with the in-flight store instruction matches a thread
4 ID associated with the load instruction.

1 24. The method of claim 21, wherein performing the dependence check further
2 comprises:

3 if the load instruction and the in-flight store instruction do not each originate with a
4 speculative thread, determining if the load instruction and the in-flight store instruction each
5 originate with a non-speculative thread.

1 25. The method of claim 21, further wherein:

2 declining to forward further comprises declining to forward the store data to the load
3 instruction if (the load instruction and the in-flight store instruction each originate with a
4 speculative thread) AND (the in-flight store instruction originates with a speculative thread
5 that is not older in program order than the speculative thread from which the load instruction
6 originates).

1 26. A method, comprising:

2 processing a speculative thread cache read request;

3 processing a speculative thread cache write request; and
4 processing a cache access request from a non-speculative thread.

1 27. The method of claim 26, wherein processing a speculative thread cache read
2 request further comprises:

3 forwarding speculative data from a cache to a speculative thread responsive to a data
4 cache read request.

1 28. The method of claim 26, wherein processing a speculative thread cache read
2 request further comprises:

3 forwarding non-speculative store data from a cache to a speculative thread responsive to
4 a data cache read request.

1 29. The method of claim 26, wherein processing a cache access request from a non-
2 speculative thread further comprises:

3 forwarding non-speculative data from a cache to a non-speculative thread responsive to a
4 data cache read request.

1 30. The method of claim 26, wherein processing a cache access request from a non-
2 speculative thread further comprises:

3 if a cache does not include a cache line associated with the cache access request, allocating a
4 new cache line;

5 wherein allocating a new cache line further comprises:

6 if the new cache line includes dirty speculative data, allocating the new cache line
7 without generating a writeback operation; and

8 if the new cache line includes dirty non-speculative data, generating a writeback
9 operation.

1 31. The method of claim 26, wherein processing a speculative thread cache write
2 request further comprises:

3 allowing the speculative thread to write data to the cache if a cache line corresponding to
4 the cache write request includes speculative data.

1 32. The method of claim 26, wherein processing a speculative thread cache write
2 request further comprises:

3 if the cache line corresponding to the cache write request contains dirty non-speculative
4 data in the cache line corresponding to the data cache request:

5 generating a writeback of the dirty non-speculative data;

6 allowing the speculative thread to write speculative data to the cache line; and

7 marking the cache line as speculative.

1 33. The method of claim 26, wherein processing a speculative thread cache write
2 request further comprises:

3 if the cache does not contain data in a cache line corresponding to the data cache address:

4 allocating a new cache line;

5 marking the new cache line as speculative; and

6 allowing the speculative thread to write speculative data to the new cache line.